PathForward Project

Procurement & Materiel
University of California /DOE
Contract W-7405-ENG-48

REQUEST FOR PROPOSAL B541520
STATEMENT OF WORK

1. Introduction

1.1 The Advanced Simulation and Computing (ASC) Challenge

The challenge undertaken by ASC to ensure the integrity of the U.S. nuclear weapons stockpile, without actual testing, is a formidable one. In order to meet this challenge, a computational modeling, simulation and analysis environment must be provided which enables unprecedented confidence in simulation results. The goal is to develop the computational equivalent of an actual experiment that simulates operation of and effects on a complete weapon system, including all subsystem components, at extremely high fidelity. The task of the Visual Interactive Environment for Weapons Simulation (VIEWS) program is to provide "see and understand" capabilities for the enormous datasets generated by these digital experiments. The results are fundamental requirements for the VIEWS program that are unprecedented in problem and analysis complexity and magnitude.

Since ASC capability-needs exceed both current and projected technology levels associated with natural evolution, ASC is consciously making efforts to accelerate technology development and commercialization where possible and appropriate. One way in which ASC is encouraging such acceleration is by providing funding to key industrial research and development activities that show promise for delivering commercially viable technologies that would be useful to the needs of ASC and VIEWS.

1.2 Increasing Visualization Needs

A critical need identified within the VIEWS program is to develop scalable rendering and visualization technologies that are matched to the massively scalable computation systems being constructed by ASC. Scalable rendering refers to the ability to build ever larger computer systems to meet the needs of ever larger datasets. As dataset sizes continue to grow, this problem will become increasingly acute. Current computational platforms deliver 30TF of computational horsepower with 100TF machines planned within the timeframe of this RFP. These computational platforms consist of thousands of nodes ranging from 2 to 16 processors per node. ASC's scalable rendering systems need to be tightly coupled to the compute platforms producing the data sets.

While parallelism is being used at many levels in today's highest performance graphics systems, when scaled to their design limits, these systems fail to deliver the performance necessary to provide desired levels of interactivity, even for current ASC data sets. This RFP seeks to provide funding for scalable rendering development efforts leading to commercial products, ensuring that appropriately scaled visualization systems, integrated with current and upcoming compute platforms, will be available for procurement from vendors in this timeframe.

1.3 Core Technologies

The ASC VIEWS program has a significant interest and investment in commercial off-the-shelf or COTS-based distributed rendering systems and has been actively deploying such systems, utilizing high-end PC graphics cards, Linux OpenGL drivers, and an Open Source software infrastructures including both the Distributed Multi-headed X11 (DMX) (dmx.sourceforge.net) and Chromium (chromium.sourceforge.net) systems. The VIEWS program requires that such systems scale to many hundreds of compute and rendering nodes as well as on the order of 50 independent, stereo capable, display devices. VIEWS is most interested in technologies that enhance and extend the ability of these technologies to meet or exceed these requirements. The need to remain cost-effective while still meeting these fundamental scalability requirements must be reiterated. Our primary concern is the software infrastructure for our existing visualization applications. This RFP is not a hardware procurement, but it is desirable that the proposed solutions make significant, if not optimal, use of the hardware rendering capabilities that exist on these deployed clusters.

2. Performance Features

2.1 Scalable rendering performance through the Chromium Application Programming Interface (API)

Existing applications are being ported to the Chromium distributed rendering API. Of key concern is the scalability of this API in both sort-first and sort-last configurations on PC clusters. A proposal must address mechanisms for defining, measuring and assuring scalability via this API. The solution must include both primitive sorting optimizations and image compositing optimizations for the applications of significant interest to the VIEWS program, including: EnSight (CEI), ParaView (Kitware) and VisIt (LLNL). Interaction with these application developers will be critical to a successful proposal.

In addition to the applications layer support, the proposal needs to address the scalability of the physical hardware layer as exploited by the rendering system. This includes features such as efficient, scalable network communication layers for Chromium, as well as “pass-through” rendering layers that ensure high-performance rendering operation on modern graphics cards.

A critical aspect of the scalable rendering solution is that it should be able to utilize a large number of resources while driving a smaller number of display tiles. In other words, when driving a tile display the rendering infrastructure should not be limited to the number of tiles. Facilities will often have 64 rendering nodes and a 4 tiled display, and the system should be able to effectively utilize the 64 nodes to drive the tiled display.

2.2 Rendering large, time varying datasets at interactive speeds

To be used interactively, the technology must be capable of generating large imagery (see point 2.3) at frame rates between 5 and 60 Hz, depending on dataset sizes and display modality. Scene sizes as large as 30 to 50 million independent triangles are common. It is important that frame rates not drop below 5 Hz even on the very largest datasets. Moreover, achieving interactive speed should not induce severe latency penalties. Image generation latency should not exceed 2-3 frames in any case.

As noted, datasets may contain hundreds or thousands of time steps. While most datasets typically contain spatial and temporal coherency there are many datasets within which little (particularly temporal) coherency can be exploited. In arriving at the frame rate requirements, it is assumed that each frame can be from a different timestep. Additionally, rendered surfaces are often dynamically generated, so technologies that rely on large geometry and texture caches must provide sufficient bandwidth and computational capacity to handle the case where the visualization primitives are recomputed every frame.

2.3. Support for both multiple offices and shared theater facilities

Scientists spend most of their time working in their offices alone or with a few colleagues. At each of the DOE weapons labs, support is needed for high-end graphics in at least 100 offices, with perhaps 20-30 concurrent users. Supporting a large number of offices from a single shared rendering facility is important. Collaborative theatre facilities using large screen tiled displays and multiple screens are also common within ASC/VIEWS. Scientists find utility in collaborative interactions supported by such facilities and the increased resolution available in such facilities is also of real utility. Current theatres use up to 48 tiles and provide display resolutions up to 64 million pixels.

The proposed technology must be capable of supporting both desktops and theatre facilities. The ideal technology would be capable of channeling all of its rendering power to either kind of environment with little or no switching overhead. In the first case, it would need to support dozens of concurrent users each exploring a different, possibly multi-terabyte dataset, on a desktop or smaller tiled display (e.g. 2x1, 2x2, etc). Dataset sizes are not necessarily reduced when lower pixel count displays are being utilized. In the second case, the technology must support a seamless high-performance display on a large tiled wall display. Thus the technology must be capable of dynamic resource allocation and resource aggregation to match dataset and display size requirements.

Stereo, both active and passive, plays a large role in the interactive investigation of datasets by VIEWS scientists. The proposed solution must be capable of rendering to multi-tiled, stereo display walls in both planar and “CAVE” configurations, with and without head tracking. The solution should provide programming interfaces to allow applications to query configuration details and modify rendering parameters accordingly to support non-planar, stereo displays.

2.4 Support for rich graphical functions and primitives

The system must support a rich palette of graphics functions and primitives including volume rendering (both structured and unstructured), and rendering of opaque and translucent polygonal surfaces. It must be possible to combine these primitives in a single generated image.

2.5 Open Source Software

The software developed under the proposed Subcontract shall be released under a University approved Berkeley Software Development (BSD) Open Source license, (see the Sample Subcontract) and delivered to the University with the source code. Offeror may propose an alternate Open Source license which would be subject to approval by the University and the DOE/NNSA.

End User documentation, component interface documentation, application programming interface (API) documentation and software test suites shall be provided as part of the open source release. The methods used for documentation and testing shall be broadly available, not requiring propriety products with limited access.

Read/Write access to the software source repository shall be provided to at least one person, designated by the University, at each of the following locations, Lawrence Livermore National Laboratory (LLNL), Los Alamos National Laboratory (LANL) and Sandia National Laboratories (SNL). Read access to the software shall be broadly available. Offeror shall describe the software gatekeeper plans.

2.6 Additional Requirements

In additional to the above the proposal shall contain the following:

- R&D or engineering plans for the proposed effort.
- A timeline for commercial product availability resulting from the proposed effort.
- Estimated cost to the University for the commercial product(s) resulting from the proposed effort.
- Plans for scalability testing of proposed technology to verify it meets the University's scalability requirements. This may include a specific request for the University to support such testing by providing access to University equipment and facilities.

3 Design Characteristics

3.1 Advanced feature sets

- Support for and exploitation of programmable graphics hardware.
- Debuggers for vertex and fragment program development.
- Increased graphics precision in compositing schemes.
- Interfaces to external input devices (e.g. immersive devices and head tracking).

In addition to the core rendering feature set outlined in the minimum requirements section, there are a number of technologies that the University would like to see available in this scalable rendering infrastructure.

The rendering system should allow a distributed, parallel graphics application to be able to exploit programmable graphics hardware. Specific support for programmable pixel shaders and other operators to accelerate common parallel operations, such as image compositing, are very desirable features. A related, desirable feature is support for interactive OpenGL fragment and vertex program debuggers that would greatly aid in the development of such programs.

Of additional interest is the support of high-dynamic range (HDR) pixel formats throughout the system. Particular support is desired for their use in large volume rendering applications where the enhanced rendering precision afforded by HDR implementations is critical for accurate visual representations.

A desirable feature would be that support for common device interfaces (e.g. trackd) be included as a supporting library (e.g. CRUT) to the core rendering system.

3.2 Supporting legacy software infrastructure

- Application transparent support for complex displays (e.g. remote network displays, tiling, etc.)
- Legacy applications support (e.g. support for single context/pipeline OpenGL based applications)

The Tri-Labs have a large body of existing software, much of which will need to be supported on clustered rendering systems. The majority of this software is written under UNIX utilizing OpenGL. While it is understood that applications will need to be modified to most effectively utilize these systems, it is desirable if the systems provide some level of legacy support and that they provide largely transparent mechanisms to simplify the necessary porting effort.

3.3 Long-term architectural viability

The use of commercial off-the-shelf components to leverage parallel technology development efforts.

- Cost effectiveness of the solution
- End user, deployment, and applications developer support paths
- Independent component upgrades, allowing the system to be customized, scaled and balanced to specific problems as well as allowing for individual components to be upgraded along different technology paths
- Platform aware, but agnostic solutions that are capable of running on a variety of platforms, interconnects, etc but can be optimized for superior performance for a given system target.

These systems will form the backbone of the Tri-Labs capability visualization services for some time to come. Solutions that offer cost effective scalability, and upgradeability are required to achieve these goals. To that end, solutions capable of leveraging emerging market technology trends and development are desired as they potentially allow a system to avoid premature obsolescence and extend the lifespan of substantial infrastructure investments.

A major concern is support for the resulting solutions, both in terms of end-user deployments and for applications developers moving to the newer architectures. A desirable component of a successful proposal would be an outline of the continuing support options that might be available for deployed systems and to applications developers.